International Journal of Computer Science Engineering
and Information Techonology Research (IJCSEITR)
ISSN (P): 2249-6831; ISSN (E): 2249-7943
Vol. 8, Issue 2, Jun 2018, 15-30

TRANS STELLAR
•Journal Publications •Research Consultancy

A CASE STUDY OF EXISTING QUALITY MODEL BASED ON DEFECTS & TESTS MANAGEMENT OF EMBEDDED SOFTWARE SYSTEM

ABHISHEK ANURAG[1] & KAMATCHI IYER[2]

[1]*Research Scholar, Department of Computer Science & Engineering, Amity School of Engineering & Technology, Amity University, Mumbai, Maharashtra, India*

[2]*Head of Department, Department of Computer Science & Engineering, Amity School of Engineering & Technology, Amity University, Mumbai, Maharashtra, India*

ABSTRACT

Any customer buying or using a product or services first thinks of quality. Quality is directly associated with what an end user expects to work and whether it's working as per requirements or features delivered. Anything not working as expected after delivery is a defect. If end- user experiences are not meeting as per his requirements, then it can be said that quality is not good; it's not meeting expectations. So, basically, quality is associated with meeting all requirements as per expectations. As per ISO 9000, quality is the degree to which a commodity meets the requirements of the customer at the start of its life. The late American Management guru Peter F. Drucker said, "Quality in a product or service is not what the supplier puts in. It is what the customer gets out and is willing to pay for" [1]. Quality can be measured in terms of whether it's meeting requirement or not, what percentage of customers are accepting or passing and what percentage of customers are rejecting or failing. If quality is not meeting its customer needs then it has defects. Defects play a major role in improving customer experiences and satisfaction. Defects detection and fix them is very crucial and so it's very critical to keep track of them in defect database from the beginning of software design and development to delivery to the customers. Defects are usually analyzed, and root caused to improve the technical and procedural aspects to help improve quality. One of analysis technique is RCA – root cause analysis of defects [2].

A case study is conducted in Organization X on an Android- based embedded system software Product Y. For the privacy reasons, organization and product names are protected in this paper. Collected data is analyzed based on Histograms, Pareto Charts, Figures and root cause analysis (RCA) is done on that. Since data record size is based on almost 21K defects all analysis is based on Microsoft Excel- based functions. A focus of this case study is defect data analysis & test data analysis and formulate a way to define the current quality model. This case study also focuses on defect data analysis using the RCA method to define root cause and corrective actions. All stages of research and case study are explained in detail based on mathematical analysis of data available.

KEYWORDS: Quality Model, Defect Management, Test Management, Data Charts, Root Cause Analysis & Corrective Actions

Received: Mar 30, 2018; **Accepted**: Apr 20, 2018; **Published**: Apr 30, 2018; **Paper Id.**: IJCSEITRJUN20183

Original Article

INTRODUCTION

Quality Project Overview

The 'Product Y' in this study is a very complex Android-based system software embedded product, which is released into the market. It's one of the world's most advanced and complex streaming device, which offers best entertainment features in Google connected environment. Total headcount for this product was 761 out of which 70 were test engineers and remaining all resources were from rest of design and development team. This product till inception to launch in the market took almost 24 months.

This product was developed based on different team's efforts and whole product was majorly differentiated into 9 major modules. There were many different hardware board types and around 926 different software components, which altogether form an architectural unified unit which is called a master module in this paper. The overall size of software products under study's complexity can be defined based on: a number of files were 588K, total lines of code were 241 million LOC and total statements 38 million LOC.

The product release, hardware and software had been very important for the United States and European market. There were 6 major releases and 10 minor releases 'over the air' (OTA) software releases were made to the customers. As part of improving end users or customer experiences and quality improvement process many retrospective processes was followed and one process is a defect root cause analysis (RCA). As part of this process all customer defects were diligently tracked in a defect database management system. Defect database management system tracked all defects originated from inception to all customer releases; defected opened by testers, developers and customers. As part of testing strategy, all tests were managed into test management database and failures against these issues were tracked as defects caught in-house. As part of the quality improvement process, a cross-functional team was formed from hardware, software different modules development team, test engineering team, integration team, certification team and quality control team. The goal of RCA was [3]:

- Analyze test coverage associated with the product under study
- Analyze defects originated from different aspects of development, testing and customers and find a systematic root cause of defects
- Map defects with test coverage
- Analyze major customer reported defects post over the air (OTA) releases made to the customers and define improvement actions

Scope of This Study

The scope of this study is to pick a very complex Android-based embedded system software product and study quality based on tests management and defect management. This case study mainly includes:

- Gather defects of the project under study from the defect database management system and analyze it's all attributes which might impact quality.
- Define different stages of software based on these defects. Associate customer defects with software stages after defect's root cause analysis (RCA).

- Gather test cases of the project under study from the test database system and analyze different aspects of test types.

Objectives of This Study

Goal of this study is to do an analysis of defects and tests and come up with:

- What are existing quality parameters available in this product?
- What is the existing quality model?
- What is the existing quality of the product under study?
- Part of this study also focuses on defining attributes of quality based on:
- End user's observations [customer defects – basically based on all non-development or non-testing defects] – convert end users issue into defects
- What is different feedback mechanism – viz. Google analytics or internal feedback mechanism or any other tools
- Defect types – categorize defect into different types associated with all aspects of root cause analysis
- Different root cause and corrective actions based on customer defects
- Identify a co-relation between: defect types, the root cause of defects, different software development stages and quality

Methodologies

Research method followed in this case study is "causal research (experiment based)". As part of study, the known dependent variable is 'Quality Score' and independent variables are:

- Phases of software development phases
- Defect types
- Root cause of defects
- Test Types

As part of this study on Product Y, the goal is to find what is quality score based on these dependent variables. Also, analyze any other variables or attributes which is going to impact quality.

Related Work

This case study is an extension of the study done under "Study on Software Quality improvement based on Root Cause & Corrective Actions". This study was also presented [Paper-id:ICETCT181055] in "International Conference on Emerging Trends in Computing Technology – ICETCT-2018, 18-19 January 2018" [4] organized in Amity School of Engineering & Technology, Amity University, Mumbai. This paper got published in International Journal of Advanced in Management, Technology and Engineering Sciences (IJAMTES).

The scope of a study done in the paper presented in ICETCT-2018 [4] was to study different aspects of software quality, defects, root causes of defects and corrective actions. The scope of this paper study was to find a current quality model and analyze other related work which is already done. Quality model is not only based on test cases execution rather there are one other major aspects of quality which is based on root cause analysis (RCA) of defects. Other related RCA methods were analyzed on how they help improve quality. The paper published was based on feasibility analysis made to create a generalized mathematical quality model which will be based on defects, the root cause of defects, corrective actions of defects and other crucial aspects of software development and testing techniques.

DATA AND ANALYSIS

Defect Data (Defect Origin)

Defect data was collected from defect database for Product Y in Company X [for sake of safety and anonymity let's call the company as X and Android-based embedded system software product as Y]. The defects were collected for period May-2013 to April-2017. Defects collected can be realized presented in Table 1.

There were many teams working on Product Y within Organization X and after analysis of defect data they were mainly grouped in 3 buckets at the high level:

- Defects logged by "Developer". There were many kinds of developers who were involved in designing, developing and testing of the Product Y and mainly called as: core engineering, dev automation, tools, team, application developers, build and infrastructure team, production team, research & white box testing team.

Table 1

May2013-April 2017	Defect Origin Teams	Count	% Count	Total
Developer	Core Engineering Team	3279	15.77%	3369
	Dev Automation Team	63	0.30%	
	Tools Team	6	0.178%	
	Application Developer Team	13	0.06%	
	Build & Infra Team	5	0.02%	
	Production Team	1	0.005%	
	Research Team	1	0.005%	
	White Box Testing Team	1	0.005%	
Tester	Core Testing Team	12094	58.16%	15030
	Test Automation Team	1778	8.55%	
	Test Certification Team	643	3.09%	
	Driver Test Team	499	2.40%	
	Hadware Test Team	14	0.07%	
	Customer Customization Test Team	2	0.01%	
Customer	End User [Non-Dev & Non-Testing]	1727	8.31%	2395
	Customer [Outside Company X]	586	2.82%	
	Marketing Team	82	0.39%	
			Total	20794

- Defects logged by "Tester". Here also many different testing teams were involved in Product Y validations and mainly called as: core testing team, test automation team, test certification team, driver/firmware validation team, hardware test team & customer specific customization validation team.
- Defects logged by "Customer". There were three types of customer. Two types among them were within Organization X itself and one who outside were 'real customers' who bought the product, used and based on bad quality, poor performance and user experiences logged the defects. Internal customers within X were: marketing team and internal end users or beta users.

Overall; out of 20794 defects majority were coming from testing & development team, but still overall customer defects were quite high and impacting quality. Total defects impact can be visualized based on Figure 1. 72% defects we caught while doing a different level of validation within the testing team. 16% of defects were caught while design, development, developer level of implementations and validation. Still, 12% of defects were from customers and these were defects which really impact user's experience and quality of the product. In this study, focus is on analyzing these customer defects on the hypothesis that their analysis will help improve quality eventually.

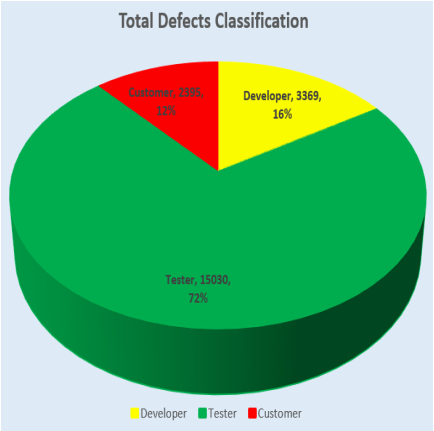


Figure 1

Defect Type

All defects gathered were analyzed to understand defects type or it's severity. When detailed analysis of defects origin was made then next level of understanding and analysis made based on type. Total 7 major defect types were identified which depicts the defect severity. These can be termed as:

- System crash – in such situation device is frozen or unresponsive or different kinds of a crash is seen viz. kernel panic. In most of such situations device is rebooted or need to power off/on to recover from the crash for the device to be responsive.

Table 2

Defect Origin		Defect Type/Severity								
May2013-April 2017	Teams	Funcationality	Task Tracking	Enhancement	Application Crash	Performance	System Crash	Corruption	Total Defect Type	Total
Developer	Core Engineering Team	1534	834	394	105	162	145	105	3279	3369
	Dev Automation Team	24	8	0	18	0	13	0	63	
	Tools Team	3	1	2	0	0	0	0	6	
	Application Developer Team	4	0	0	2	3	1	3	13	
	Build & Infra Team	5	0	0	0	0	0	0	5	
	Production Team	1	0	0	0	0	0	0	1	
	Research Team	1	0	0	0	0	0	0	1	
	White Box Testing Team	0	0	0	0	0	0	1	1	
Testing	Core Testing Team	7765	1025	656	1201	516	317	614	12094	15030
	Test Automation Team	628	250	235	184	37	402	42	1778	
	Test Certification Team	610	19	4	5	0	3	2	643	
	Driver Test Team	389	7	7	24	41	5	26	499	
	Hadware Test Team	11	0	0	1	0	0	2	14	
	Customer Customization Test Team	1	0	1	0	0	0	0	2	
Customer	End User [Non-Dev & Non-Testing]	1048	162	207	101	75	65	69	1727	2395
	Customer [Outside Company X]	324	121	46	22	27	33	13	586	
	Marketing Team	50	5	12	7	4	1	3	82	
									Total	20794

- Application crash – Application(s) being used on device is frozen or crashed viz. ANR (application not responding error message) is coming on the device or in logs or application crash is seen. In this situation mostly,

an application is not responding, but the device is still responsive.

- Functionality – functional aspects of a device are not working viz. wi-fi is not working. If any functionality is the broken customer is not able to use that feature and it creates a panic among customers. This kind of defects is maximum seen by either developer or tester or customers.
- Corruption – There are many kinds of corruption viz. audio is having noise or glitch, video or streaming is not proper with desired quality, image or video displayed on display panel is not clear, the files created and stored are not usable as file format is corrupted, etc. Corruption causes customer audio/visual experiences to go down drastically and impacts quality.
- Performance – these are all kinds of events which makes device usage go slow. Viz. Apps downloads, and installation time is very high while launching any games, it's taking time more than expected, audio/video latency – audio and video are not reaching on time and delay is creating users to wait to listen or see the content.
- Enhancement – when customer usage the device, then based on the device or application of features, usage, suggests improving or add new features for his better experience. Viz. the device is only having Wi-fi support and the user is asking to have ethernet support also.
- Task Tracking – It's a kind of defect which tells what kind of activity or tasks are being worked upon based on its priority and importance. Viz. for any enhancement requests opened a test engineer might open a task tracking defect for test design & planning activity.

Table 2 captures complete statistical analysis for: defect origin, different teams. Different teams are consolidated in terms of developer, tester & customer. Also, different defect types are analyzed and identified based on their severity.

Based on defect type analysis, it was observed that most customer encounters issues related to functionality of features delivered. One or another way the functionality is broken for users or not working as per required. And, so enhancement requests were high from them and it caused task tracking activities to be increased to help resolve such issues.

Another view of customer defects and its overall type could be understood better in terms of what % of defects are of which type. In Table 3 we could very well be analyzed and understood that ,out of total 12% defects from customers almost 7% of total defects are of functionality type and to help improve quality these needs to be addressed.

Based on data in Table 3 it was also analyzed and observed that 'system crash, application crash performance and corruption' related customer defects were less than 1% each compared to overall defects. So, most of the time whenever a software release was made to customers based on new features implementations, defect fixes; existing functionality got broken as new regressions popped up [here regression is something which was working earlier and now it's not working].

Table 3

Defects Type Vs. Defects Origin	Developer	Tester	Customer	Customer Defect Type (%)
Funcationality	1572	9404	1422	6.84%
Task Tracking	843	1301	288	1.39%
Enhancement	396	903	265	1.27%
Application Crash	125	1415	130	0.63%
Performance	165	594	106	0.51%
System Crash	159	727	99	0.48%
Corruption	109	686	85	0.41%
Total	**3369**	**15030**	**2395**	**12%**

Defect Priority

Another attribute which defines the impact of customer experience and quality is defect priority. As part of development, design, implementations, validation & releases made to customers; defects logged always carry a priority to understand it's impact and what is the next course of action to fix them. It's very necessary to understand the priority and impact of defect as it's associated with experience and quality. As part of this study, all defects were categorized into different priorities based Product Y under study. The priority definition is carried across all products of organization X.

Total 4 different defect priorities are as below in which all defects [Table 4] were put:

- U [Unprioritized] – these are defects on which no one is working to resolve. These might not be defects which are of due importance for product owners, release managers, developers, testers or customers. No one is actively considering such defects for closure and impact analysis.
- P1 [Showstopper – a release blocker] – these are defects which are of the most important ones. These defects will immediately cause a customer device to go down or unusable. These defects are critical enough to block a major portion of critical testing for test engineering team. If a software release needs to make to the customer, these kinds of defects will stop the release immediately. Unless these defects are not fixed release cannot be shipped.

Table 4

Defect Origin		Defect Priority						
May2013-April 2017	Teams	U	P1	P2	P3	P4	Total Defect Priority	Total
Developer	Core Engineering Team	1486	11	1427	276	79	3279	3369
	Dev Automation Team	21	0	32	10	0	63	
	Tools Team	1	0	2	3	0	6	
	Application Developer Team	3	0	10	0	0	13	
	Build & Infra Team	0	0	5	0	0	5	
	Production Team	1	0	0	0	0	1	
	Research Team	0	0	1	0	0	1	
	White Box Testing Team	0	0	0	1	0	1	
Testing	Core Testing Team	1052	29	7090	3551	372	12094	15030
	Test Automation Team	63	2	1372	300	41	1778	
	Test Certification Team	5	10	596	29	3	643	
	Driver Test Team	22	1	321	149	6	499	
	Hadware Test Team	3	0	7	3	1	14	
	Customer Customization Test Team	1	0	1	0	0	2	
Customer	End User [Non-Dev & Non-Testing]	0	1	843	639	244	1727	2395
	Customer [Outside Company X]	0	0	197	350	39	586	
	Marketing Team	0	0	43	19	20	82	
							Total	20794

- P2 [Fix before next build] – these are defects which are not severe enough to block a release or customer device to become unusable. But, if such defects are seen by the customer they will log a high priority issue and will keep asking frequent questions & updates till defect is not resolved and new software is released to them. If such defects are observed within developers or testers it slows down their productivity and efficiency of the product. These defects are tried to resolve in next upcoming build to ensure customer or tester or developers are not

impacted highly.

- P3 [Fix before next milestone] – these kinds of defects are annoy able to customers or anyone using product. But, they might not escalate for frequent updates or keep asking when defect will be resolved. Such defects are planned to fix in upcoming major milestones or depending on what kind of impediment tester or developer is facing. These kinds of defects are picked to be resolved once P1 & P2 defects are addressed as customers are more interested in them.
- P4 [Nice to have] – these defects are a kind of wish list for better usage and comfort of customers. These don't annoy anyone but if resolved makes them happy and comfortable.

After the setting priority of defects and doing further analysis it's found that majority of defects falls under P2 [fix before next build] or P3 [fix before next milestone] – and these impacts almost 83% of defects [Figure 2]. When analyzed only customer defects as part of this study then it came out that even P2 & P3 defects logged by customers are 87% [Figure 3], which is comparable based on overall defects and only for customer defects. Developer and tester defects also majorly fall into P2 & P3 categories of defect priorities.

This is also comparable with defect types in Table 2 and as per that defect types: functionality (60%), application crash (8%), system crash (5%), performance (4%) and corruption (4%) constitutes 81% of defects and in align with P2 & P3 priorities [for total defects it's 83% and for customer defects it's 87%] of defects.

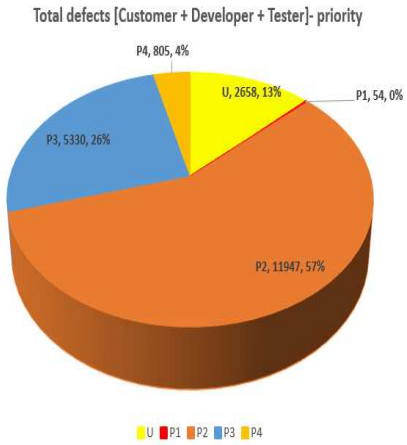


Figure 2

There are two other aspects of defects which were not analyzed as part of this study:

- Defect disposition
- Defect regression

This analysis will be done in the extension of this study and will be captured as part of that. We can consider these as limitations of this study of time being. So, in terms of defects 3 aspects were captured and analyzed here is: 'defect origin', 'defect type' & 'defect priority'.

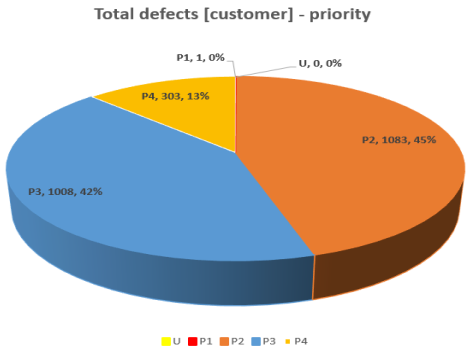


Figure 3

Test Data

As part of this study, all tests available into the tests database management system were analyzed and identified that what are tests applicable to Product Y which is under study. Table 5 represents all test data gathered and analyzed for the period May-2013 to April-2017. After analysis, it was found that out of 33.7K tests 67% tests were not valid for Product Y validation and not taken into consideration for different verification stages. 33% of tests were valid ones, out of that 22% were manual ones and 11% were automated ones. This product was validated based on both manual and automated tests execution.

For sake of anonymity and privacy of confidential data; major area or modules name is also not disclosed here. There we mainly 10 major modules in which whole tests were distributed. Out of that 4 modules were not applicable on Product Y due to technical nature and requirement of the product.

Table 5

Major Area	Android Based Product Tests (May2013-April 2017)			
	Invalid Tests	Valid (Manual)	Valid (Automated)	% Automation
Module 1	6158	0	0	0.00%
Module 2	521	0	0	0.00%
Module 3	1885	4	13	76.47%
Module 4	541	0	0	0.00%
Module 5	1599	753	320	29.82%
Module 6	784	244	470	65.83%
Module 7	4421	1990	1297	39.46%
Module 8	4597	3635	1543	29.80%
Module 9	2068	799	92	10.33%
Module 10	165	0	0	0.00%
Total	22574	7425	3735	33.47%

Test Type Definition

In this study, all valid tests were analyzed to understand what kind of tests are being executed in different major area or modules. Based on this analysis first, different tests type and its definition were created. Table 6 captures test types and definition based on approximately 11k tests which were valid for a product under study.

As part of this study it was not focused on doing classifications of 11k tests based on these test types [7].14 different test types were defined [Table 6]. As part of the extension of this study all valid 11k tests will be categorized under ‘test type’ definition provided in Table 6and it’s mainly based on ISO 26262compliance and leveraged here for this

study. For the time being we can consider this as a limitation of this study. Need to further analyze how these tests and test types are mapped with different attributes of 'defect' and quality.

While doing this analysis, multiple cross-functional team were created based on people who were part of this product. Mainly based on different defect origin teams of developer, tester & customers, software stages, product managers, technical marketing team and quality experts.

Table 6

Test Type	Definition
Requirement Tests	Tests mapped on software requirements [based on Functionality, Features, POR/design]
Interface Tests	Tests mapped on software requirements [based on different component interface]
Fault Injection Tests	Tests mapped on injecting faults or issues or error or negative values. Mainly negative tetsing
Resource Usage Tests	Tests mapped on checking different resources, their capacity limits, concurrent usage
Code Coverage Tests	Tests mapped to line coverage, function coverage, statement coverage, branch coverage, call coverage, decision making
Boundary Tests	Tests mapped with values near boundaries or out of those boundaries
Certification Tests	Tests mapped to get certification
Stress Tests	Tests mapped to validate system under stressful conditions and for long duration
Stability Tests	Tests mapped to validate system reliability under normal conditions and for long duration
Perf & Power Tests	Tests mapped to validate performance and power consumption of system
Usability/OOBE Tests	Tests mapped to validate end user experinces
Manufacturing Diagnostic (factory) Tests	Tests mapped to valiadte factory flow of hardware products
Design Test	Tests mapped to validate software and hardware design specifications
Product polish Tests	Tests mapped to validate how polished product is in terms of customer experiences

Root Cause Analysis

As part of root cause analysis, all customer defects [2395 defects in Table 2] was studied, analyzed and brainstormed for its root cause and possible corrective actions. As part of RCA process at different level, multiple groups and forums had been formed and all defects were analyzed. The team formed for discussion were based on:

Table 7

Root Cause Category	Definition	Corrective Actions
Invalid [not actionable defect]	These defects are invalid - no action can be taken [not actionable]	No Action - mostly inactionable comments from customers
Duplicate of tester defect	Test engineer has already opened defect for same issues	No Action - as tests exist to catch such defects and are part of test coverage
Test exists [execution missing]	Test cases exist but was not executed	Add existing test cases into test covearge
Test exists [tests steps modification]	Existing test cases need modifications for test steps or methods or expected results or test environment or setup	Modify existing test cases
Request for enhancement [RFE]	Request for enhancement for existing or new feature	Design new test cases based on new features implemented or existing features modified
Test doesn't exist [tester level]	No existing test cases to catch such defects	Design new test cases to catch such defects
Test doesn't exist [dev level]	Need a unit level test/script/apk or test methods from developer to catch such defects	Design new test cases to catch such defects with help of developer at dev level tetsing or at test engineers level
Test doesn't exist [customer HW specific]	No existing test case to catch such defects on internal hardware. I will be caught only on customer specific hardware [test environement or setup]	Design new test cases to catch such defects on customer specific hardware [replicate customer setup or environment or configuration]
Customer specific customization	Customer specific tools, test apps and/or code implementation	Design new test cases based on customer specific tools, test apps and/or code changes

- Defects origin team [Table 2] – experts were picked from 17 different teams except from customer group. Customers were from outside organization so only members of the internal team were chosen for analysis.

- Module leads or subject matter experts from each module team [Table 5]. Here experts were from development teams, test engineering team, technical marketing team and internal users of Product X.
- Subject matter experts involved in different SW phases; 'Requirements gathering & analysis, design, implementation or deployment, verification or validation or testing and maintenance' were involved in the analysis.
- Cross-functional team was created from different products of a similar kind mainly Android-based system software products, product leads, program managers, release managers, members from technical marketing and sales & marketing team; all were having representations in the analysis of such customer defects.

Majorly root cause categories can be discussed, understood, agreed and approved are as below:

- Invalid – these customer defects are not actionable. There is no meaningful information provided by customers in defects raise so no action can be taken inside the organization.
- Duplicate of test engineer's defect – while doing validation a test engineer had already logged defects which root cause is exactly like customer defects of this type. When customer defects were analyzed, and root caused then it was found that similar issues are already being seen and reported inside development or testing team.

Table 8

Root Cause Category Vs. SW Stages	Requirements Gathering & Analysis	Design	Implementation	Verification	Maintenance	Total [Root Cause Category wise]
Invalid [not actionable bugs]	118	1	488	537	69	1213
Duplicate of tester bug	4	0	73	43	7	127
Test exists [execution missing]	13	2	146	37	37	235
Test exists [tests steps modification]	3	0	81	15	11	110
Request for enhancement [RFE]	111	2	129	48	5	295
Test doesn't exist [tester level]	10	4	143	27	25	209
Test doesn't exist [dev level]	0	0	37	25	12	74
Test doesn't exist [customer HW specific]	1	0	14	13	8	36
Customer specific customization	6	0	66	22	2	96
Total [SW Phase wise]	266	9	1177	767	176	Total Customer Defects = 2395

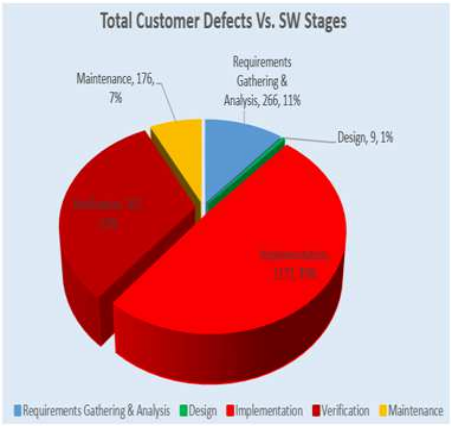
- Tests not executed – tests are there in the test database management system, but due to one or another reason were not executed before a customer release on release build. If these existing tests were executed the defects raised would be caught in-house.
- Test doesn't exist – no test is there in test database which would have been able to catch defect. Either new tests need to be designed or existing tests need to be modified. It might also have needed to modify the test environment or setup or configuration within which tests needs to be executed.
- Request for enhancement – a customer is asking for new features or existing feature needs design modifications. When a customer starts using the product, and delivered features he finds that either it needs modification or need some supporting new features which would have improved his experiences.
- Customer specific customizations – there might be a very specific need to the customer and if that were implemented but it caused defects. Here also new tests need to be designed to catch such defects.

Based on these root causes major corrective actions were analyzed and its mainly:

- Execute existing tests for every customer release
- Design new tests or modify existing tests

All customer defects were analyzed to map with root cause category and in which SW phase it is mapped with where it would have been originated and caught. The complete mapping can be seen in Figure 4. After analysis,it was concluded that 49% defects could have been caught at the implementation stage and 32% at validation stage. So, a total of 81% defects was at implementation and validation stage and out of those defects, 26% [Table 8 – Test exists & doesn't exist] of total defects could have been caught just by executing existing tests, designing new tests and modifying existing tests.

Figure 4



SW phases identified mainly areas: Requirements Gathering & Analysis, Design, Implementation or Deployment, Verification or Validation and Maintenance [5, 6].

Existing Quality

For the Product Y which is under study, a release Y which was made to the customer was taken for study and analysis here for quality aspects. A quality report was made for internal quality assessment for making release decision. Release Y had a milestone quality target and quality assessment was based on all valid tests executed; manual and automated tests both. For every module there were multiple test plans based on test types and grouped together to tell quality of each module. It was basically tracked on ways of how many tests are executed, how many tests are passed and how many tests are failed or blocked. All instances of test results were tracked in the test database. Every failed test has defects logged into the defect database management and based on business, product and release priority defects different attributes are set, reviewed and agreed.

In this Release Y [Table 9], pass % of overall product based on all applicable modules was 93.07%. Remaining tests were either failing or blocked [tests were not able to be executed due to defect]. A release was still made to customers based on:

- Tests passed 93.07% were more than 90% [within acceptable range of 90% to 95%].
- Defects logged based on 6.93% tests failed or blocked were scrubbed with the help of all stakeholders and found that none of the defects associated were in priority P1 [showstopper – release blocker] category.

- Defects having priority P2 [fix before the next build] and P3 [fix before next release milestone] were analyzed and observed that none of the defects have a probability to be escalated by customers as P1 [Showstopper].

CONCLUSIONS

In this paper, a Product Y of Company X in conjunction with Release Y were analyzed for quality to determine existing software quality model. All known attributes of quality were

Table 9

Major Area	Android Based Product Tests Status (May2013-April 2017)					
	Invalid Tests	Valid (Manual)	Valid (Automated)	% Automation	Pass % [Release Y]	Target Pass %
Module 1	6158	0	0	0.00%	NA	NA
Module 2	521	0	0	0.00%	NA	NA
Module 3	1885	4	13	76.47%	60.00%	95%
Module 4	541	0	0	0.00%	NA	NA
Module 5	1599	753	320	29.82%	99.74%	95%
Module 6	784	244	470	65.83%	99.59%	95%
Module 7	4421	1990	1297	39.46%	93.26%	95%
Module 8	4597	3635	1543	29.80%	92.51%	95%
Module 9	2068	799	92	10.33%	87.18%	95%
Module 10	165	0	0	0.00%	NA	95%
Total	**22574**	**7425**	**3735**	**33.47%**	93.07%	95%
Pass % Legend	< 90%	>= 90% & < 95%	>= 95%	NA = Not Applicable		

Taken into consideration and data were gathered from test database and defect database. The different attributes which are having characteristics to impact quality were identified and are as below:

Defect Database

- Defect origin
- Defect type
- Defect priority
- Defect root cause
- Defect corrective actions

Test Database

- Test types
- Test module

Software Stages

- Requirements Gathering & Analysis
- Design
- Implementation
- Verification
- Maintenance

All these attributes are independent variable and quality score is a dependent variable. Pass% is Quality-Score and is being studied as the dependent variable. In this study, it was observed that in Company X, Product Y; the only attributes based on which a quality is defined is "Pass %". Other aspects of defects & tests were not taken into consideration for quality model. Another attribute which was taken into consideration to make Release Y was defect priority, none of defects have priority as P1 [showstopper – release blocker] while making a release and none of P2 & P3 priority defects have potential to be P1 eventually. But, this aspect was not considered in the mathematical quality model of the product Y under study.

Technically and mathematically in this existing model the attributes taken into consideration are:

- Total number of tests execution
- Total number of tests pass
- Total number of tests fail or block

Mathematically; Quality-Score is defined for Company X & Product Y is as:

Pass % = Quality-Score = [total number of tests passed / total number of tests executed]

Limitation and Future Work

- Few major attributes of the defects were not analyzed. While making this study it came into picture that these attributes have the potential to impact quality of software and they might have weighted contribution in new quality modeling. In this study attribute of customer defects which were not studied and analyzed are:

a. Defect regression

b. Defect disposition

c. Defect module

- The co-relation between all attributes and their weight on quality were not analyzed. Correlation and weighted impact needs to define for:

a. **Defects**– "origin, type, priority, regression, disposition, module, root cause and corrective actions"

b. **Tests** – "types"

c. **SW stages** – "Requirements Gathering & Analysis, design, implementation, verification & maintenance"

- Code Coverage – what is the impact on quality based on: "total number of lines of Code (LOC) in the product, Number of lines executed while doing validation, dead code, number of lines cannot be tested. Doing a feasibility analysis of adding code coverage in association with rest of attributes and their impact on quality.

Future study, will focus on analyzing above limitations, identify a correlation between all attributes identified and create a mathematical new quality model (mostly weighted model based on attributes overall impact). It will also focus on doing the case study of implementing the model and analyzing results on products of Company X and in other companies too.

ACKNOWLEDGEMENT

A sincere thanks to Prof. (Dr.) R. Kamatchi, Professor, Head, CSE, Amity School of Engineering and Technology, Mumbai; being mentor and guide and for continuous help and support to come up this paper.

REFERENCES

1. *https://www.lifetime-reliability.com/cms/free-articles/work-quality-assurance/what-is-quality/*
2. *CaglaAtagoren; Information Technology and System Management, Bas¸ kent University, 06810, Ankara, Turkey e-mail: caglaatagoren@gmail.com. OumoutChouseinoglou; Statistics and Computer Science Department, Bas¸ kent University, 06810, Ankara, Turkey e-mail: umuth@baskent.edu.tr. A Case Study in Defect Measurement and Root Cause Analysis in a Turkish Software Organization. R. Lee (Ed.): SERA, SCI 496, pp. 55–72. DOI: 10.1007/978-3-319-00948-3_4 c Springer International Publishing Switzerland 2014*
3. *Marek Leszak; Lucent Technologies, Optical Networking Group, Thurn-und-Taxis-Str. 10, 90411 Nuernberg, Germany. Dewayne E. Perry; Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX. Dieter Stoll; Lucent Technologies, Optical Networking Group, Thurn-und-Taxis-Str. 10, 90411 Nuernberg, Germany: A case study in root cause defect analysis Proceeding. ICSE '00 Proceedings of the 22nd international conference on Software Engineering Pages 428-437 ACM New York, NY, USA ©2000 table of contents ISBN:1-58113-206-9 doi>10.1145/337180.337232*
4. *Abhishek Anurag, Ph.D. Student in Computer Science & Engineering, Amity School of Engineering & Technology, Amity University Mumbai & Guide: Prof. (Dr.) Kamatchi Iyer, Head, CSE, Amity School of Engineering and Technology, Mumbai: "Study onSoftware Quality improvement based on Root Cause & Corrective Actions" (Paper-id: ICETCT181055). International Journal of Advanced in Management, Technology and Engineering Sciences, IJAMTES/319, ISSN No.: 2249-7455, IJAMTES Journal, Volume 8, Issue IV, April – 2018.*
5. *Rani, Deevi Radha, and S. Venkateswarlu. "Software and hardware defence methods against cache-based side channel attacks." Int J Manage Inform Technol Eng 2.5 (2014).*
6. *Ejaz, Amna, and Asfandyar khalid. "Effectiveness of Requirements Elicitation Techniques in Software Engineering Process: A Comparative Study based on Time, Cost, Performance, Usability and Scalability of Various Techniques."*
7. *Roger Pressman; Software Engineering – A Practitioner's Approach; Tata McGraw Hill, Sixth Edition, 2010*
8. *Pankaj Jalote's Software Engineering – A Precise Approach, Wiley Precise, First Edition, 2010*
9. *A quick guide to ISO 2622, Feabhas Ltd, 5 Lowesden Works, Lambourn Woodlands, Hungerford, RG177RY, Www.feabhas.com, info@feabhas.com @ https://www.feabhas.com/sites/default/files/2016-06/A%20quick%20guide%20to%20ISO%2026262%5B1%5D_0_0.pdf*